PROJEKT – TBO

Czas trwania:, termin oddania 29.01.2023
Punkty do zdobycia: 30

Opis

Celem zadania jest zasymulowanie środowiska, w którym rozwijana jest aplikacja wykorzystująca proces CICD, w którym zaimplementowane zostały mechanizmy bezpieczeństwa mające ochronić przed stworzeniem aplikacji, która zawiera podatności bezpieczeństwa. Każdy zespół, który składa się z 1 do 3 osób będzie miał do zrealizowania dwa zadania:

- Zaprojektowanie i zaimplementowanie procesu CICD, który będzie zawierał kroki polegające na uruchomieniu testów typu SAST, testów typu SCA oraz testów typu DAST, a następnie wzięcia pod uwagę wyniku tych testów przed zbudowaniem obrazu z aplikacją
- Udowodnienie, że ten proces działa poprawnie poprzez próbę wstrzyknięcia kodu zawierającej jakąś podatność. Proces CICD, w którym realizowane są testy bezpieczeństwa powinien wykryć problem we wprowadzonej zmianie i zablokować wdrożenie aplikacji.

Środowisko i przygotowanie

Projekt można zrealizować zarówno na platformie github.com jak i gitlab.com - w zależności od preferencji zespołu. Zarówno rozwiązanie CICD, które zostanie wykorzystane przez zespół jest w pełni dowolne np. Gitlab-ci, github actions, travisCI.
Należy pamiętać, że infrastruktura, na której działa CICD zarówno na gitlab.com i github.com a która należy do providera jest częściowo darmowa pod warunkiem tego, że projekty są publiczne i czas zajętości runnerów jest ograniczony. Istnieje alternatywa polegająca na wykorzystaniu gitlab.com oraz zainstalowaniu runnera na swoim komputerze a następnie zarejestrowanie go w swoim projekcie w gitlabie.
Aplikacja, która będzie przedmiotem projektu – czyli, dla której proces CICD będzie realizowany jest również dowolna. Może to być jakaś aplikacja, która była realizowana w ramach innego przedmiotu ale mogą to też być przykładowe aplikacje, które były dostępne w zadaniach laboratoryjnych.
Zespół musi wybrać:

- Gdzie będzie przechowywany kod źródłowy (np.. Github.com)
- Aplikacja, dla której będzie budowany proces CICD (np.. Aplikacja JAVA z laboratoriów)
- Jakie rozwiązanie CICD będzie wykorzystywane (np. Github actions)

Zadanie 1 – zaprojektowanie procesu CICD

Zadanie polega na zbudowaniu procesu CICD, oraz skonfigurowaniu repozytorium w taki sposób, że:

1. Obraz aplikacji z tagiem latest budowany jest z wybranej gałęzi np. Main
2. Pull Requesty / Merge Requesty lub bezpośrednie pushe do repozytorium może wykonywać wyłącznie właściciel repozytorium
3. Nowe gałęzie może tworzyć dowolny użytkownik

4. Wypchnięcie zmiany do nowej gałęzi ma skutkować uruchomieniem procesu cicd, który zbuduje obraz dockerowy o tagu :beta
5. Przed zbudowaniem jakiejkolwiek wersji proces CICD musi uruchomić: testy jednostkowe, testy SCA, testy SAST oraz testy DAST

Zadanie 2 – Weryfikacja działania procesu CICD

Zadanie polega na zweryfikowaniu czy zaimplementowany proces działa poprawnie. Aby wykonać weryfikacje należy:
1. Przygotować nową gałęź kodu, w której dodana zostaną 2 nowe podatności bezpieczeństwa - mogą to być podatności typu wstrzyknięcia kodu, podatności które wykorzystują podatność w bibliotece opensource lub inne - ważne żeby faktycznie można było zaprezentować, że kod w nowej gałęzi zawiera podatność bezpieczeństwa która prowadzi do wystąpienia incydentu. Czyli - jeśli dodano lub zmieniono kod tak aby można było wykorzystać podatność SQL Injection ta podatność powinna być możliwa do wykorzystania, analogicznie w przypadku podatności w bibliotekach opensource, nie wystarczy dodać do pliku pom.xml lub requirements.txt biblioteki, która zawiera podatność - ta podatność musi być możliwa do wykorzystania
2. Zweryfikować, czy proces CICD uruchomiony na nowej gałęzi się nie powiedzie (tutaj czytaj uwagi w kolejnej części)

Uwagi i notatki

- Do zadania 1: potencjalne rozwiązania skanujące bezpieczeństwo umieszczam w załączniku na końcu tego dokumentu
- Do Zadania 1: dobrze byłoby aby proces CICD kończył się wypchnięciem zbudowanego obrazu do rejestru obrazów dockerowych docker.io - w przypadku publicznych repozytoriów kodu możliwe jest stworzenie publicznego rejestru dockerowego, który jest bezpłatny https://www.docker.com/pricing/
- **Do zadania 1: Celem zadania jest stworzenie procesu CICD. Jeśli wybrałeś aplikacje która zawiera podatności bezpieczeństwa (np. Z laboratoriów) celem zadania nie jest usunięcie tych podatności - nie przejmuj się nimi, jak są wykrywane wszystko jest ok.**
- Do zadania 2: Jeśli spróbujesz wstrzyknąć bibliotekę z podatnością np. Log4j ale się nie uda ponieważ proces CICD nie pozwoli na takie dodanie – to też jest ok, opisz dlaczego dodany przez Ciebie fragment kodu jest złośliwy i jak można by go wykorzystać np. Screen z aplikacji uruchomiony u siebie na stacji lokalnej - niektóre procesy realizujące testy podatności w procesie CICD są niezwykle skuteczne
- **Do zadania2: jeśli proces CICD już przed przystąpieniem do zadania wykrywał podatności bezpieczeństwa, weryfikacji podlega to, czy po wprowadzeniu zmiany konkretne problemy, które zostały w wyniku testów - sprawdzamy diff'a**

Wyniki projektu

Wyniki projektu muszą być przedstawione w formie repozytorium. Proces CICD musi być zaimplementowany w pliku .gitlab-ci.yml lub w pliku .github/workflow oraz opisany w pliku readme.md.

Zadanie drugie polegające na próbie wstrzyknięcia złośliwego kodu zrealizowane ma być jako oddzielna gałąź w repozytorium, na której uruchomi się zadanie CICD.

Oceniane będą: repozytorium, definicja CICD oraz readme.md dla zadania 1 oraz nowa gałąź, zmodyfikowany w niej kod oraz wynik zadania CICD dla nowej gałęzi dla zadnia 2

Wyniki podesłane mają być via Teams w odpowiedzi na zadanie projektowe. Odpowiedz musi zawierać:

1. Listę osób w zespole
2. Link do projektu w gitlab lub github (proszę nadać uprawnienia użytkownikowi @siewer)
3. Link do wyniku zadania CICD, dla zadania 2 (zakładam, że w repozytorium tych jobów może być dużo)

Skanery bezpieczeństwa

Przykładowe skanery bezpieczeństwa:

- SCA: OWASP Dependency check https://github.com/jeremylong/DependencyCheck
- SAST: python: bandit - https://pypi.org/project/bandit/ java: spotbugs https://spotbugs.github.io
- DAST: OWASP ZAP: https://www.zaproxy.org